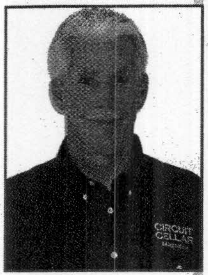
by Tom Cantrell (USA)



Wall Whisperer

A Fresh Take on Powerline Communication

Powerline communication isn't a new concept, but as the Rodney Dangerfield of networks, it just can't get respect. So, maybe your experience with early versions was less than satisfying. Now Cypress is ready to give it another go. How about you?

"Home Automation" is seemingly an oxymoron, one of those "Tomorrow's Technology of Tomorrow" that just can't get off the starting block. I've got a relatively new home, and it's about as "smart" as a rock with just a few brain cells (e.g., a thermostat and a garage door opener).

Half the battle is the skeletons in the closet. Like a lot of you, I had my fling with X10 many years ago. And no doubt, like a lot of you, it didn't last long. The performance and reliability was just too iffy for me to convince myself to put a lot of work into trying to do something meaningful.

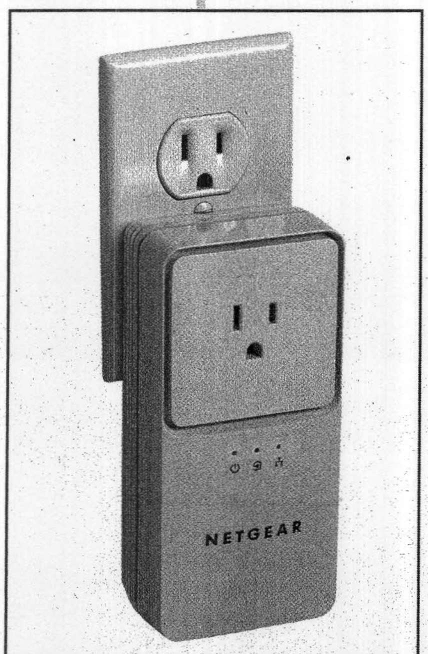


Photo 1—Powerline communications is making a comeback with a new generation of high-speed units such as the 200-Mbps Powerline 200 AV+ from Netgear.

The concept of powerline communication (PLC for short) took another hit with the misguided premise of turning your power company into an ISP. The concept of "Broadband over Power Line" (aka BPL) may have sounded good at the time, but as a practical matter, the idea it can compete with DSL and cable is fading away.[1] With its dubious track record, I haven't really bothered to keep track of PLC. But taking off the blinders, I can see that there are some interesting developments afoot. You'll see what I mean if you Google acronyms like "P1901," "G.hn," and "UPA."[2] Or toddle over to your nearest BUY MORE big box and check out the latest PLC offerings from networking heavyweights like Linksys and Netgear (see Photo 1). Their newest powerline gadgets claim to deliver up to 200 Mbps, with 500 Mbps and even 1 G on the drawing board. While it's generally understood these claims greatly overstate actual throughput, it's still fast by any measure. Beyond some metering and "smart grid" applications, BPL may be dead as far as your power company is concerned, but the concept would seem to have much better prospects in smaller scale applications.

This is where Cypress Semiconductor steps in with three new PLC-capable chips. Let's take a closer look and see how they let embedded applications hitch a ride on the power line. Heck, the wires are already there, so why not use them?

The Cypress lineup comprises three different part numbers—the CY8CPLC10, CY8CPLC20, and CY8CLED16P01—that target somewhat different applications. As an interesting aside, the silicon under the hood is the same for the trio—namely, a standard PSoC, with differentiation provided using its programmable hardware and software capabilities. Let's start with the CY8CPLC20 (see Figure 1), which is the most general-purpose of the trio.

The PLC20 is simple enough to describe: it's a PSoC flash memory MCU—specifically, a

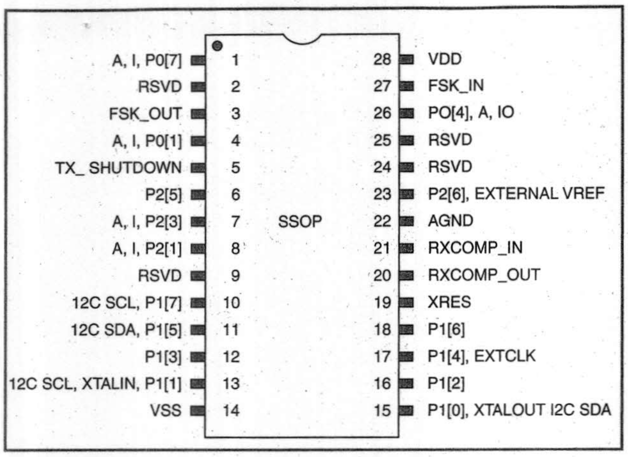


Figure 1—With the same silicon under the hood, the new Cypress powerline communication chips have a similar pinout and are available in 28-pin (shown here) and 48-pin versions.

PSoC-1 (i.e., M8C core)—whose programmable hardware is configured with a power-line modem (see Figure 2), optionally fronted with a bit of software implementing a simple network protocol.

Now before you get too excited thinking about hooking up your HDTV, realize this modem is a simple half-duplex 2,400-bps unit with performance that harkens back to the dial-up days. It's speedy enough for a lot of simple and useful applications. Just be careful not to bite off more bandwidth than the chip can chew.

When you checkout the power line transceiver (PLT) object from the PSoC Designer library, there are three options to choose from: basic PLT, PLT + network software, and PLT + network software + I2C.

The basic PLT option gives you raw byte-at-a-time access to the data link, much like a UART. The automatically generated API allows configuring the modem (e.g., bit rate, gain, noise threshold), checking if the line is busy (i.e., voltage higher than configured threshold), and transmitting and receiving bytes by interrupt or polling. With this option you'll have to handle all the network administration and error handling yourself. It's probably OK for simple and relatively static applications, but if the network configuration is large and dynamic (i.e., number of nodes unknown and changing), consider using the "PLT + network stack" option.

Behind the Cypress network stack is the fairly sophisticated packet structure shown in Figure 3. Let's go through it to see some of the embellishments the stack provides, starting with addressing. You can see the network has plenty of elbow room with 8-, 16- and 64-bit address options, the latter taking advantage of the unique 64-bit ID each chip comes with from the factory. There are also "group" and "broadcast" options that support one-to-many and many-to-one associations.

The "Service Type" bit up front is an innocuous name for a key feature of the protocol, namely acknowledgement. When enabled, receivers will automatically acknowledge receipt, transmitters will timeout in half a second if the acknowledgment isn't returned and you can configure the stack for up to four automatic retries.

After the address is a "Command" byte. Sixteen commands are defined (Table 1) with another 32 reserved and the rest available for application use. After the command are "Payload Length" (0 to 31 bytes), a sequence number (to detect lost/duplicate packets) and a header CRC. These are followed by the payload itself and an overall packet CRC.

The media access strategy implemented by the stack features "carrier sense" and "collision avoidance. Each node waits until the line is idle, but then further waits a random amount of time (between 85 and 115 ms) before initiating transmission. Without this "random backoff," every node would try to jump in as soon as the line goes idle and the network would get paralyzed by collisions.

Of course, there's no free lunch. The price for the network stack's reliability and ease of use may be worth it, but there is a price to pay—namely, throughput. You can see that in the worst case (64-bit addresses) transferring a single byte payload would require sending a 22-byte packet (i.e., a 22:1 packet/payload ratio). Fortunately, many applications will be able to use shorter addresses and/or longer payloads to reduce the overhead. For instance, with 8-bit addresses and 31-byte payloads, the packet/payload ratio would approach 1:1 (e.g., 1.2 for a 38-byte packet carrying a 31 byte payload). On the other hand, keep in mind that intermittent interference will be more likely to disrupt a longer packet than a shorter one.

That brings us to the PLC10 chip, which utilizes the final "PLT + network stack + I2C" option. In this case, access to the PLT and network stack is rerouted via the external I^2C interface. You don't have any direct access to the on-chip processor.

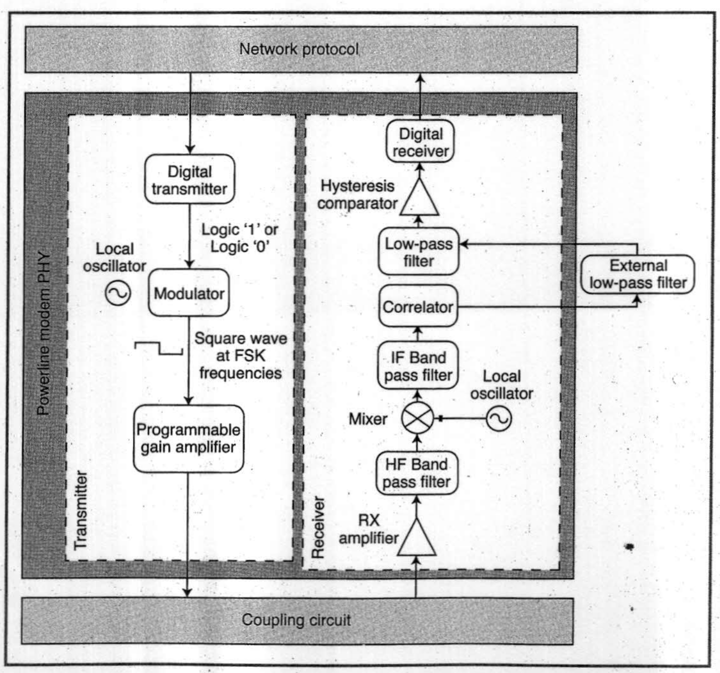


Figure 2—Cypress takes advantage of the PSoC's programmable hardware to implement a 2,400-bps, half-duplex FSK powerline modem.

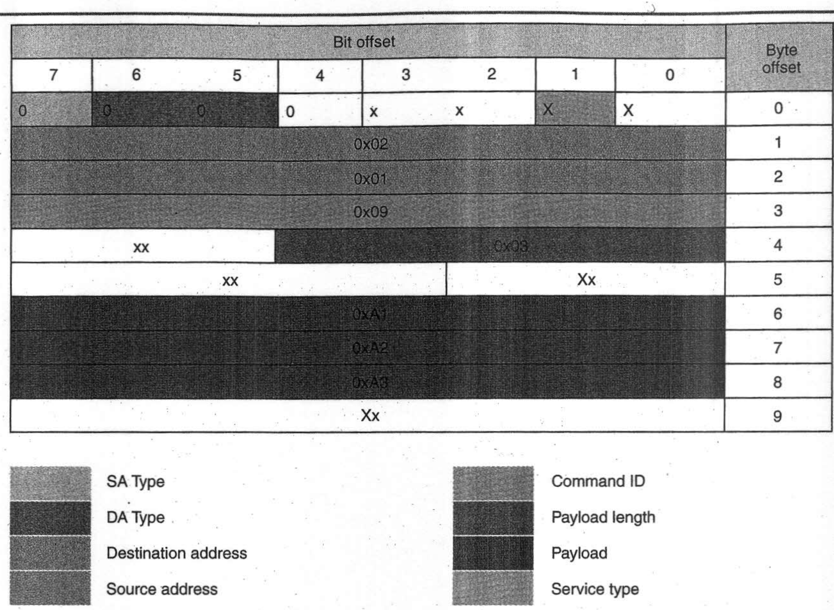


Figure 3—The Cypress powerline communicaton stack raises the level of network abstraction utilizing the packet structure shown here.

Rather the chip acts a self-contained powerline "coprocessor" working under the direction of an external MCU of your own choosing.

Although most PLC headlines revolve around 120/240-VAC, the Cypress solution is just as well-suited for lower voltage AC and DC powerline designs. Enter the CY8CLED16P01, which targets LED lighting applications, drawing from a library of hardware and software functions such as dimming, temperature compensation, color mixing and closed loop optical feedback. Another low voltage application that comes to mind would be vehicles where data would ride on the 12-VDC power bus. Just the thing for moving some bits bumper-to-bumper and everywhere in-between (e.g., using lighting and accessory power connections) without stringing a bunch of extra wire.

Command ID	Command Name
0x01	SetRemote_TXEnable
0x03	SetRemote_ExtendedAddr
0x04	SetRemote_LogicalAddr
0x05	GetRemote_LogicalAddr
0x06	GetRemote_PhysicalAddr
0x07	GetRemote_State
0x08	GetRemote_Version
0x09	SendRemote_Data
0x0A	RequestRemote_Data
0x0B	ResponseRemote_Data
0x0C	SetRemote_BIU
0x0D	SetRemote_ThresholdValue
0x0E	SetRemote_GroupMembership
0x0F	GetRemote_GroupMembership

Table 1—The network stack option eases application software development by providing a number of high-level commands.

AC/DC

"Eavesdropping aliens and the industrial archaeologists of the future may well wonder why we built a phased array across the north of England to beam credit card numbers and digital images of naked ladies into the atmosphere."—Nick Long of the Low Power Radio Association (www.lpra.org).

These days, silicon is so advanced that most designs involve little more than slapping down a few Wunderchips like Lego blocks. And so it is with the Cypress chips which handle all manner of applications with aplomb. But when it comes to actually getting the data on and off a high-voltage powerline, there's still a need for some old-school circuitry to deal with unique safety and regulatory issues. Fortunately, Cypress has an excellent app note that addresses the subject in great detail.[3] Here are some of the highlights.

Making "The last inch" connection involves three functions:

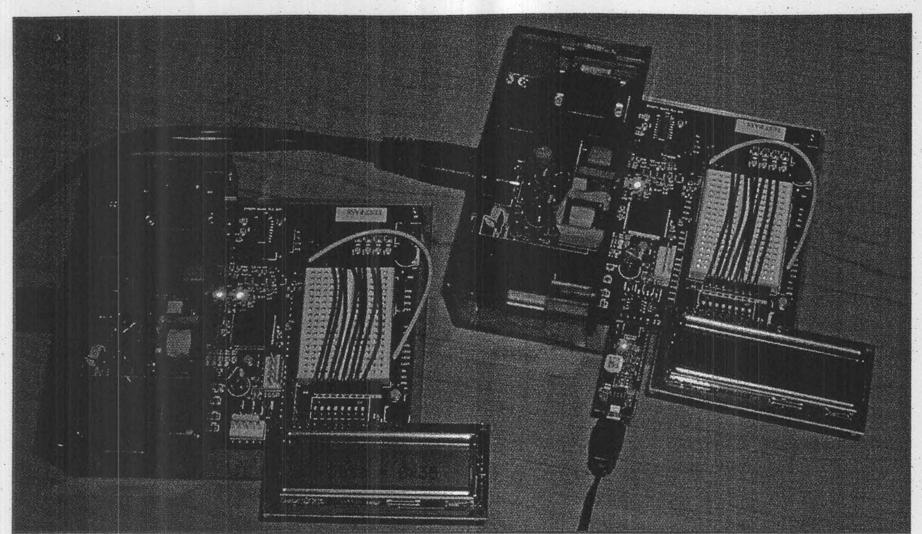
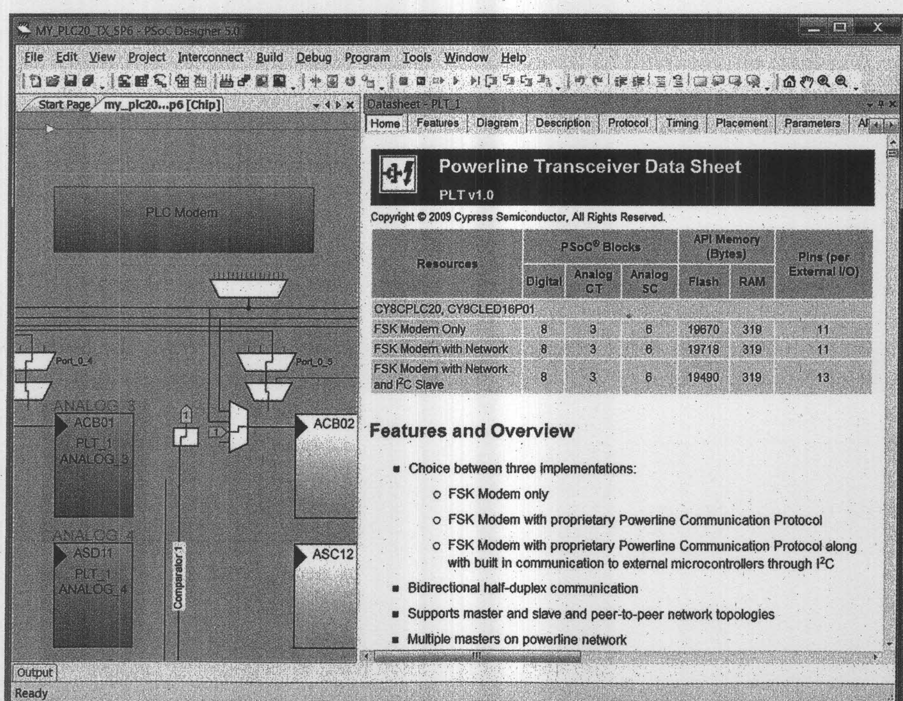


Photo 2—A pair of CY3274 kits makes it easy (if not cheap at $624 each) to evaluate the new Cypress powerline communication chips. The kit includes a programmer for downloading code and an I²C "wiggler" (attached to the board on the right) for exercising the chip's PLC10 I²C co-processor mode.

transmit filtering and amplification, receive filtering, and electrical coupling. And, of course, there's also the usual power supply circuits (e.g., 120/240 VAC to 5-V DC switcher) to power the Cypress chip itself along with any other digital logic.

Put it altogether and the powerline interface turns out to be the tail wagging the chip. A picture is worth a thousand words and Photo 2 shows the Cypress high-voltage CY3274 development kit. More on the kit in a moment; but for now, just note the nontrivial collection of transformers, op-amps, resistors, capacitors, diodes and such (more under the board) required to power-up your data.

Before moving on, I should say a few words about one controversial aspect of PLC alluded to in the quote above—namely, potential pollution of



Resources	PSoC® Blocks			API Memory (Bytes)		Pins (per External I/O)
	Digital	Analog CT	Analog SC	Flash	RAM	
CY8CPLC20, CY8CLED16P01						
FSK Modem Only	8	3	6	19670	319	11
FSK Modem with Network	8	3	6	19718	319	11
FSK Modem with Network and I²C Slave	8	3	6	19490	319	13

Photo 3—The Cypress chips aren't really "new." They're standard PSoCs upgraded with powerline communication features using the chip's programmable hardware and software capability, all under the purview of the familiar PSoC Designer tool.

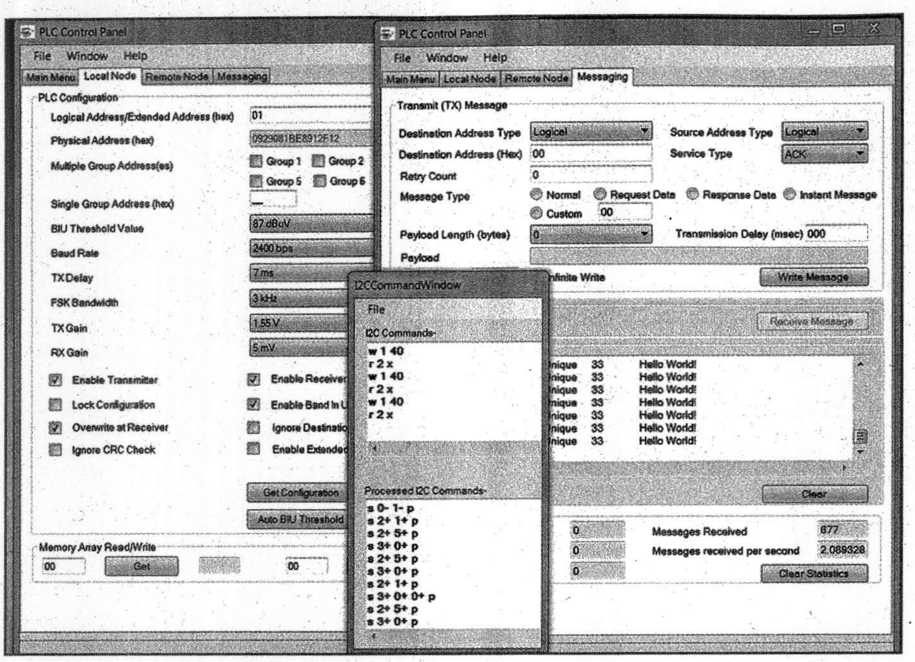


Photo 4—Configured as a PLC10, access to the powerline is via an I²C interface. The kit includes "PLC Console" software that utilizes the included I²C "wiggler" to evaluate that capability.

the radio spectrum. Amateur radio operators have been raising the red flag for years and have kept regulatory agencies' (e.g., FCC) feet to the fire.

Presumably, the demise of the full-fledged BPL concept (i.e., power company as ISP) will provide some solace for those concerned. Hopefully, limiting the reach (i.e., between walls vs. overhead lines) will mitigate the spectrum pollution concerns as well. Cypress is quick to point out their design is compliant with a variety of emission, immunity, and safety regulations and the fact they've incorporated specific features (e.g., transmit filtering) to try to be a good neighbor.

Now, as we've seen on Wall Street, the fact something "meets regulations" doesn't mean it's not going to cause problems if the regulations and/or regulators are inept. But that's not a "technical issue," it's a political one. Those who are concerned are fully entitled to stand up for their rights, but the place to hold the protest rally is Washington, DC, not Silicon Valley.

SHOCK IT TO ME

Hopefully, the clever phrase you just read will get your attention. Needless to say, if you're going to be playing around PLC, don't! "Playing around" isn't something you do with a mains voltage. It's OK to poke willy-nilly at digital logic, because if something goes wrong, the chips lose. Make a mistake with 120/240 VAC and you lose.

Turning our attention back to the Cypress CY3274 development kit, another "shocking" aspect becomes apparent, namely sticker shock. These puppies (mains- and low-voltage versions are available) are a whopping $642 each not to mention that you need at least two of them to do anything. Bracketing the CY3274 kit are some simpler, and less costly ($299) demo boards and at the top end ($951) a CY8CLED16P01 kit with LED daughter card.

The price is high, but at least the CY3274 kit is replete with features. The built-in breadboard and LCD make for instant out-of-the-box gratification and easy prototyping. Recalling the difference between the PLC20 and PLC10 chips is a matter of programmability, the kit can support development for both.

For developing PLC20 apps you utilize the popular and well-proven PSoC Designer tool, with hardware and software libraries updated to support the powerline transceiver module (see Photo 3). The kit includes a USB plug-in programmer to burn your own code into the PSoC, noting that a full-fledged debugger is an extra cost item.

Using the programmer, you can download the aforementioned "PLT + network stack + I2C" configuration to make the PLC20 on the board act like a PLC10 co-processor under the direction of an external host via an I²C connector on the edge of the board. A very nice touch is that the kit also includes a USB-to-I²C wiggler and powerline control panel software for the PC. With two kits and their wigglers you can plug into two power outlets, two USB ports and run two copies of the control panel to exercise the network from the comfort of your PC (see Photo 4).

The quick-start guide has you wire up the LCD and then program each board with a simple PLC20 application, one board as a transmitter and one as a receiver. When you push a button on the transmitter it reads the on-board DIP switch and sends the data to the receiver, which in turn displays the received data on its LCD.

I'm all for keeping it simple, but Cypress could have made this demo a little more useful. As an initial experiment, I wanted to do a quick site survey of my house wiring. Exercise is a good thing, but the prospect of running up and down stairs pushing the switch on the transmitter and then checking the receiver, for each outlet no less, wasn't appealing.

Fortunately the CD in the kit has all the demo projects and software (see Listing 1). It's testimony to PSoC Designer that even though I haven't used the tool in awhile, it was a no-brainer to tweak the demo code to have the transmitter repeatedly send a unique message without having to push the switch. Now I could make just make one pass through the house with the receiver in hand to check reception at each outlet.

Next, I reprogrammed the boards to act like PLC10s and used the two I²C wigglers with two copies of the control panel software to get a little deeper under the hood. Notably, the control panel allows you to tweak key parameters, such as gain and noise threshold, to see their effect.

So how'd it all work out? Quite well, I'd say. But note that an exhaustive and authoritative test would be a much bigger job. So let me just tell you what

I saw, but as well what I didn't see (i.e., issues a real-world designer would have to question, test and design around).

Right off the bat, you'll have to deal with the reality that there's good chance that two particular outlets won't be able to talk to each other. At least that's true for locales that use "split phase" distribution, which brings, for example, 240 V to the panel and then splits it into separate 120-V legs. Sure enough, my simple walk-around "Can you hear me now?" test confirmed some half the outlets were inaccessible from the other half. Interestingly, and a bit of good news, was the fact that the legs seemed rather well intermixed—for example, between floors and side-to-side in my two story house. There are a lot of AC outlets in a modern house—maybe four to six per room (which is a big advantage over the alternative of piggybacking on phone wires and coax)—and it generally seemed that if one outlet was dead a nearby one worked. But your mileage may vary, and unless you've got a wiring diagram, your safest option is to try before you buy.

A related gotcha is that all outlets aren't created equal, the irony being a "better" outlet in terms of power delivery may be worse for communication. Sure enough, plugging my test setup into a UPS with surge protector confirmed it was real good at killing ones and zeros as well as surges. That's something to consider if a particular installation is prone to lightning or otherwise destructive power spikes. And what about power failures? Interestingly, I don't see any reason communication couldn't continue which isn't the case for schemes, such as X10, that rely on zero crossing. Of course, there's the catch-22 that since you can't use a UPS each gadget would need a battery, a dubious solution counter to the fundamental premise. Better just figure if the power goes off, so does you application.

Presuming you've got plain-vanilla outlets that can see each other, the next question is how well the link works. Show typical house wiring, with its ad hoc layout, un-terminated stubs, power strips, extension cords, etc. to a network guru and they would laugh out loud. The topology is very inhospitable and things only get worse when you consider all the sources of noise such as motors, fluorescent lights, dimmers, and so on.

While the Cypress network stack brings a lot of features to the table, it doesn't do everything. There is one API call that automatically tests the line noise level and sets the noise threshold for determining if the line is "busy." But beyond that your own software will have to handle features such as adaptive gain settings. Another embellishment would be security with the understanding that signals may leak between properties that share a transformer.

Next I used PSoC Designer to further tweak the demo software to track packet errors and display the count on the LCD. I set up the units at opposite corners of the house (bottom floor left, top floor right) and let it run overnight. Checking next morning I discovered the unit had detected a grand total of 10 errors, about one per hour. That's quite impressive considering I was sending an 8-byte packet (8-bit addressing, 1-byte payload) about once per second, making for an overall packet error rate of less than 0.01%.

Switching back to the PLC Console mode (i.e., using the I^2C wigglers) confirmed similarly good results, at least with a simple one-way test, but digging a little deeper revealed some issues. On the receiver side, somewhere along the line between the I^2C connection, USB and the PLC Console

Listing 1—Taking advantage of the Cypress network stack API makes it easy to send and receive data via the powerline. Shown here is the receive routine utilized by the demo software that comes with the CY3274 kit.

```
void main()
{
   BYTE bTemp;
   LCD_Start(); // Start the LCD Module
   PLT_Start(); // Start the Powerline Transceiver
   TX_LED_Start();RX_LED_Start();BIU_LED_Start();   // Start the powerline comm. LEDs
// Set the baud rate to 2400 bps with a 3 kHz deviation
   PLT_Memory_Array[Modem_Config] = (Modem_TXDelay_7ms|Modem_FSKBW_3M|Modem_BPS_2400);
   PLT_Memory_Array[TX_Gain] = 0x0b; // Set the TX Gain to 0.375x
   PLT_Memory_Array[RX_Gain] = 0x00;
// Set the logical address of the PLC20 Receiver board to 0x01
   PLT_Memory_Array[Local_LA_LSB] = 0x01;
// Enable the TX and RX and allow the RX buffer to be overwritten
   PLT_Memory_Array[PLC_Mode] = (TX_Enable|RX_Enable|RX_Override);
   LCD_Position(0,0); LCD_PrCString("PLC20 RX");
   while (1)
   {
     PLT_Poll(); // Run the network protocol to update memory array
     bTemp = PLT_Memory_Array[INT_Status];
     if ((bTemp & Status_RX_Data_Available) && (PLT_Memory_Array[RX_CommandID] ==
     CMD_SENDMSG) && (PLT_Memory_Array[RX_SA] == 0x02))
// Process the next block of code only after a new message is received
     {
       LCD_Position(1,0); LCD_PrCString("RX Data = 0x");
       LCD_PrHexByte(PLT_Memory_Array[RX_Data]); // Display the data received on the LCD
       PLT_Memory_Array[RX_Message_INFO] = 0x00; // Reset the RX Message Info variable
     }
   }
}
```

GUI things were getting bogged down to the point that the receiver couldn't keep up with even a few incoming messages per second. I noticed the little "CPU tachometer" widget on my PC screen was redlining, something that almost never happens. Without turning on the software option that allows received packets to be overwritten—or, alternatively, the option to insert a delay between packets on the transmitter side—the receiver couldn't keep up.

Finally, I performed a pseudo-full-duplex stress test by having the nodes send and receive messages to each other simultaneously, a scenario which caused the number of errors to skyrocket. One explanation might be that the collision avoidance scheme isn't sophisticated enough to deal with such blatant abuse. But it just as well could be issues with the PC/I²C/GUI lash-up as described above.

Clearly, a real evaluation of performance and reliability will be better served with a dedicated test configuration. For a PLC20, that would be native code test routines running on the internal PSoC, while for the PLC10 it would be diagnostic software on an external MCU. Bottom line, my experience is a reminder to keep your expectations in check understanding performance is limited, the link isn't perfect, and errors will occur.

ONE WIRE WORLD

When it comes to mains powerline communication, the Cypress solution is definitely a step up from something like X10. Yes, it isn't perfect, but thanks to the closed loop acknowledgement capability at least you'll know when a packet is lost. Detected errors are a chore, but undetected ones are a showstopper.

Although less obvious, I'm thinking the sleeper opportunity is non-mains applications such as low-voltage lighting, vehicles, and indeed anything with a power cord.

Power, and data, to the people!

Tom Cantrell has been working on chip, board, and systems design and marketing for several years. You may reach him by e-mail at tom.cantrell@circuitcellar.com.

REFERENCES

[1] J. Leydon, "Powerline Networking Firm Suffers Brownout," *The Register*, 2002, www.theregister.co.uk/2002/03/28/power_line_networking_firm_suffers/.

[2] V. Oksman and S. Galli, "G.hn: The New ITU-T Home Networking Standard," *IEEE Communications Magazine*, 2009.

[3] J. Hushley, "Cypress Powerline Communications Board Design Analysis," 001-55427, Cypress Semiconductor, 2009, www.cypress.com/?docID=18145.

SOURCE

CY8CPLC10, CY8CPLC20, and CY8CLED16P01 Powerline communication chips
Cypress Semiconductor | www.cypress.com

NEED-TO-KNOW INFO

Knowledge is power. In the computer applications industry, informed engineers and programmers don't just survive, they *thrive* and *excel*.

For more need-to-know information about topics covered in Tom Cantrell's Issue 240 article, the *Circuit Cellar* editorial staff highly recommends the following content:

—

Power Analysis Primer
From Powerline Measurements to PFC
by Robert Lacoste
***Circuit Cellar* 229, 2009**

Familiarize yourself with PFC. Robert explains how to measure the power consumption of line-powered devices. Topics: Powerline, PFC, Load, Power Measurement

Go to: www.circuitcellar.com/magazine/229.html

—

Power Line Modems Meet Home Control
by Brian Millier
***Circuit Cellar* 142, 2002**

A poorly designed control system prevented Brian's heat recovery ventilators from being effective. So, he tried powerline tech. Topics: Powerline, FSK, Control

Go to: www.circuitcellar.com/magazine/142toc.htm